Research Heaven,
West Virginia

World's most civil servant

World's worst title

Martha: a next generation testable language

Not a "language" but AI agents to help analysts explore a range of model behaviors

Tim Menzies
West Virginia University
tim@timmenzies.net

A change of view

Research Heaven,
West Virginia

- **As is**
  - Requirements, analysis, design
    - Explore options
  - Code
    - Enshrine the decisions
- **To be**
  - Don't "code" but "codes"
    - Scribble down the options
    - Run them, see how they behave
  - Deliver the system
    - AND the operations manual
    - AND the work-arounds
  - E.g. ISS issue logs: "n" denotes known issues with work-arounds:
    <1, 1n, 2, 2n, 3, 4, 5>
- **But how to encode the options?**

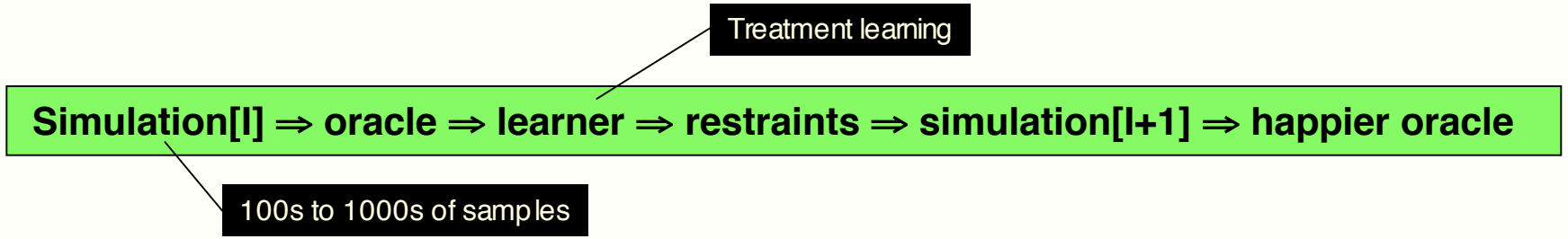
The “best language” myth

Lesson: there is no one best language
Case study: from LURCH to SPY
Best we can hope for is to cover range of possible languages

Case studies

Research Heaven,
West Virginia

- **NEAR: "language" = influence tables**
- **SILAP: "language" = functional network**
- **DDP: "language" = semantic net**
- **In all cases:**
  - Languages have sentences
  - Languages execute
  - Execution scored by oracles
  - Sentences have choice points
  - Choice points selected from distributions
  - Data mining find distributions restraints that make the oracles happier

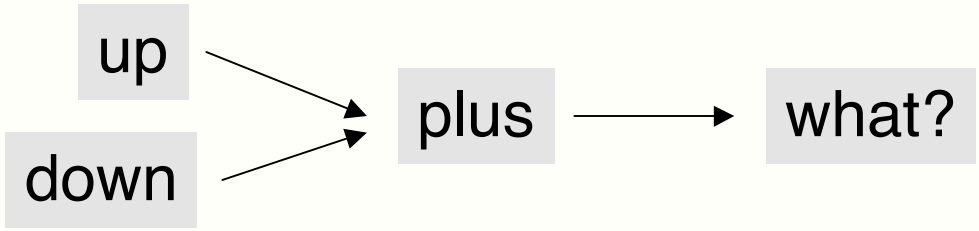


Research Heaven,
West Virginia

Case study 1

NEAR: “language”= influence tables

(work with Dr. Julian Richardson, RIACS)



Tables of qualitative influences

id	software process option	safety	dev. time	dev. cost	life cycle cost	capability
1	target critical mission phases	+	+	+	-	-
2	target critical commands	+	+	+	-	-
3	target critical events	+	+	+	-	-
4	onboard checking	+	-	-	+	0
5	reduce flight complexity	+	+	+	?	-
6	test fly prototypes	+	+	+	?	?
7	enhance safing	+	-	-	+	?
8	certification	+	?	?	?	?
9	increase vv	+	-	-	+	?
10	reduce onboard autonomy	?	+	+	-	-
11	reuse across missions	?	+	+	?	?
12	increase developer capabilities	+	+	+	?	?
13	increase developer tool use	+	+	+	?	?
14	implement optional functions after launch	?	+	?	?	?
15	reduce vv cost	0	0	+	+	0
16	increase vv speed	0	+	0	0	0
17	increase vv capabilities	+	+	+	0	+

Nondeterminant: "-" and "+" = what?

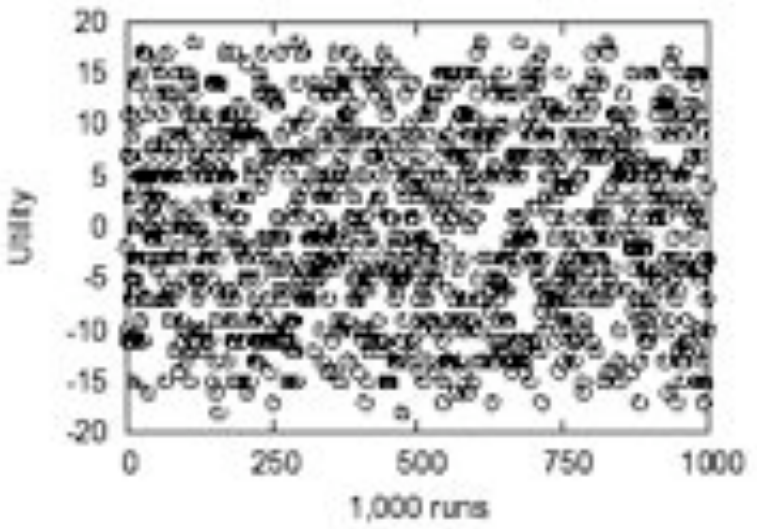
Monte Carlo to sample model structure

Research Heaven, West Virginia

- **Utility = sum(Goal(X) * Impact(Option,X))**
- **Goals:**
  - 0 <= Safety <= 10
  - 0 <= DevTime <= 10
  - 0 <= DevCost <= 10
  - 0 <= LifeCycleCost <=10
  - 0 <= Capacity <= 10
- **Er… what structure?**

id	software process option	safety	dev. time	dev. cost	life cycle cost	capability
1	target critical mission phases	+	+	+	-	-
2	target critical commands	+	+	+	-	-
3	target critical events	+	+	+	-	-
4	onboard checking	+	-	-	+	0
5	reduce flight complexity	+	+	+	?	-
6	test fly prototypes	+	+	+	?	?
7	enhance safing	+	-	-	+	?
8	certification	+	?	?	?	?
9	increase vv	+	-	-	+	?
10	reduce onboard autonomy	?	+	+	-	-
11	reuse across missions	?	+	+	?	?
12	increase developer capabilities	+	+	+	?	?
13	increase developer tool use	+	+	+	?	?
14	implement optional functions after launch	?	+	?	?	?
15	reduce vv cost	0	0	+	+	0
16	increase vv speed	0	+	0	0	0
17	increase vv capabilities	+	+	+	0	+

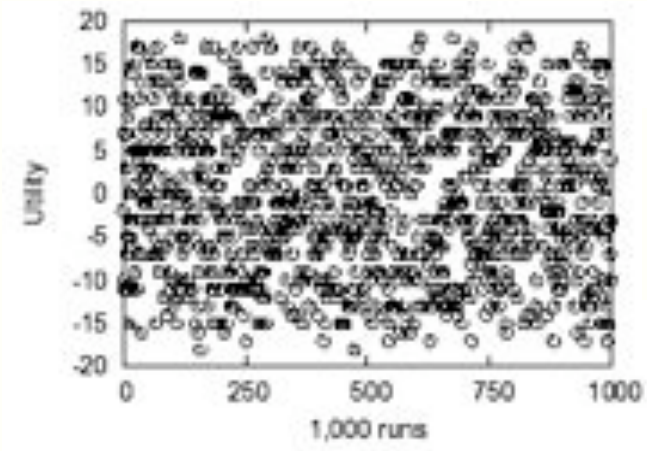
Nondeterminant: "-" and "+" = what?



Seek the “lift” structure

Research Heaven,
West Virginia

- **Divide scores into low, medium, high (scored 2,4,8)**
- **Baseline= (2 * # lows) + (4 * # mediums) + (8 * # highs)**
- **lift(attribute=range) = log((all ∩ range)/baseline)**
  - Lift = 0 if useless
  - Lift > 0 if useful
  - Lift < 0 if dangerous
- **Often, a few outstanding ranges**



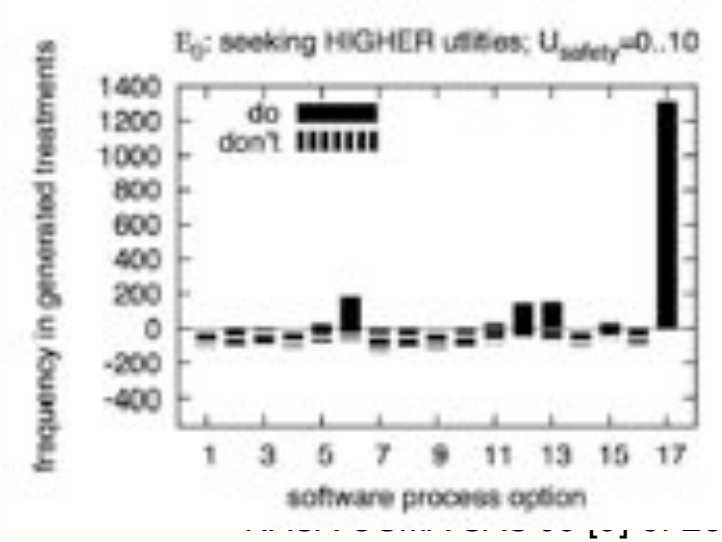
=>

lift1	frequency
-3:	[1]
-1:	[1]
0:	[59]
1:	[20]
2:	[5]
3:	[6]
4:	[1]

Build “super lifters”

Research Heaven,
West Virginia

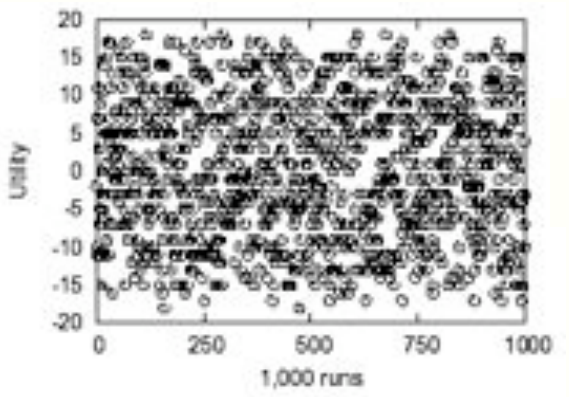
- **“Super lifter” = combinations of ranges with high lift**
- **Found = ()**
- **Interesting = 5**
- **While Interesting**
  - 100 times do
    - Build treatments using ranges with higher lifts (selected randomly)
  - Best = top 20
  - if Best in Found then
    - Interesting --
  - Else
    - Interesting = 5
    - Found = Found + Best



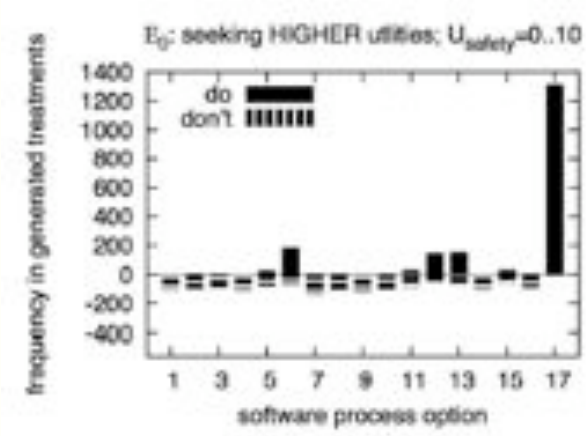
Apparent noise + treatments = improvement

Research Heaven, West Virginia

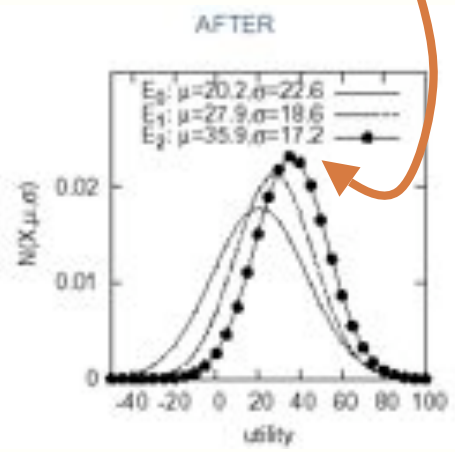
Mean utility doubles (20 to 35.9)



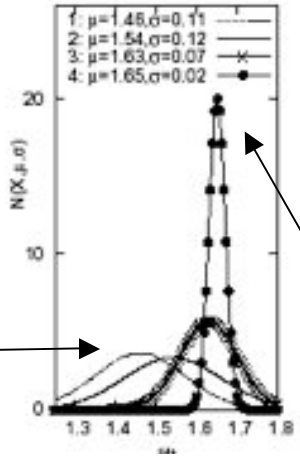
+



=



Q: But what about the variance?
A: larger and larger treatments
= less and less variance



1 lifters

4 lifters

Research Heaven,
West Virginia

Case study 2

SILAP: “language”= functional network

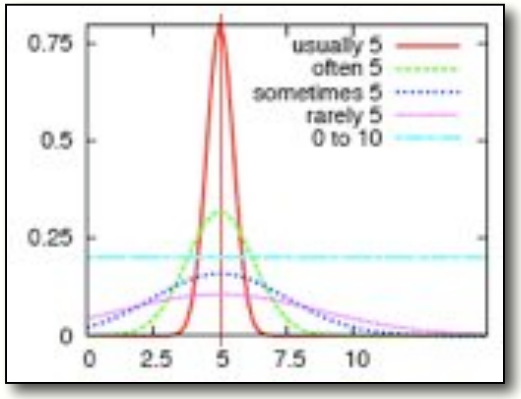
(work with Marcus Fischer, IV&V)

```
All["Experience"                ] = 0.8 to 0.9   ;
All["SoftwareDevelopment"       ] = 0.4 to 0.7;
All["Complexity"                ] = often  0.547 ;
All["Innovation"                ] = often  0.351 ;
All["SoftwareProcess"           ] = 0.1 to 0.4 ;
All["Maturity"                  ] = often  0.242 ;
All["Reuse"                     ] = often  0.226 ;
All["Organization"              ] = often  0.172 ;
All["SoftwareCharacteristics"] = often  0.172 ;
All["FormalReviews"             ] = often  0.1119;
All["SoftwareSize"              ] = often  0.102 ;
All["ConfigManagement"          ] = often  0.0962;
All["Standards"                 ] = often  0.0955;
All["DefectTracking"            ] = often  0.0873;
All["CMM"                       ] = often  0.0764;
All["RiskManagement"            ] = often  0.0647;
```

```
One["Experience"        ] = 4;
One["Organization"      ] = 2 to 4;
One["Complexity"        ] = usually 2;
One["Innovation"        ] = usually 4;
One["SoftwareSize"      ] = usually 2;
One["Standards"         ] = often 2;
One["ConfigManagement"] = sometimes 5;
One["CMM"               ] = sometimes 1;
One["FormalReviews"     ] = rarely 3;
One["DefectTracking"    ] = rarely 3;
One["RiskManagement"    ] = rarely 3;
One["Reuse"             ] = rarely 3;
One["Maturity"          ] = rarely 3;
```

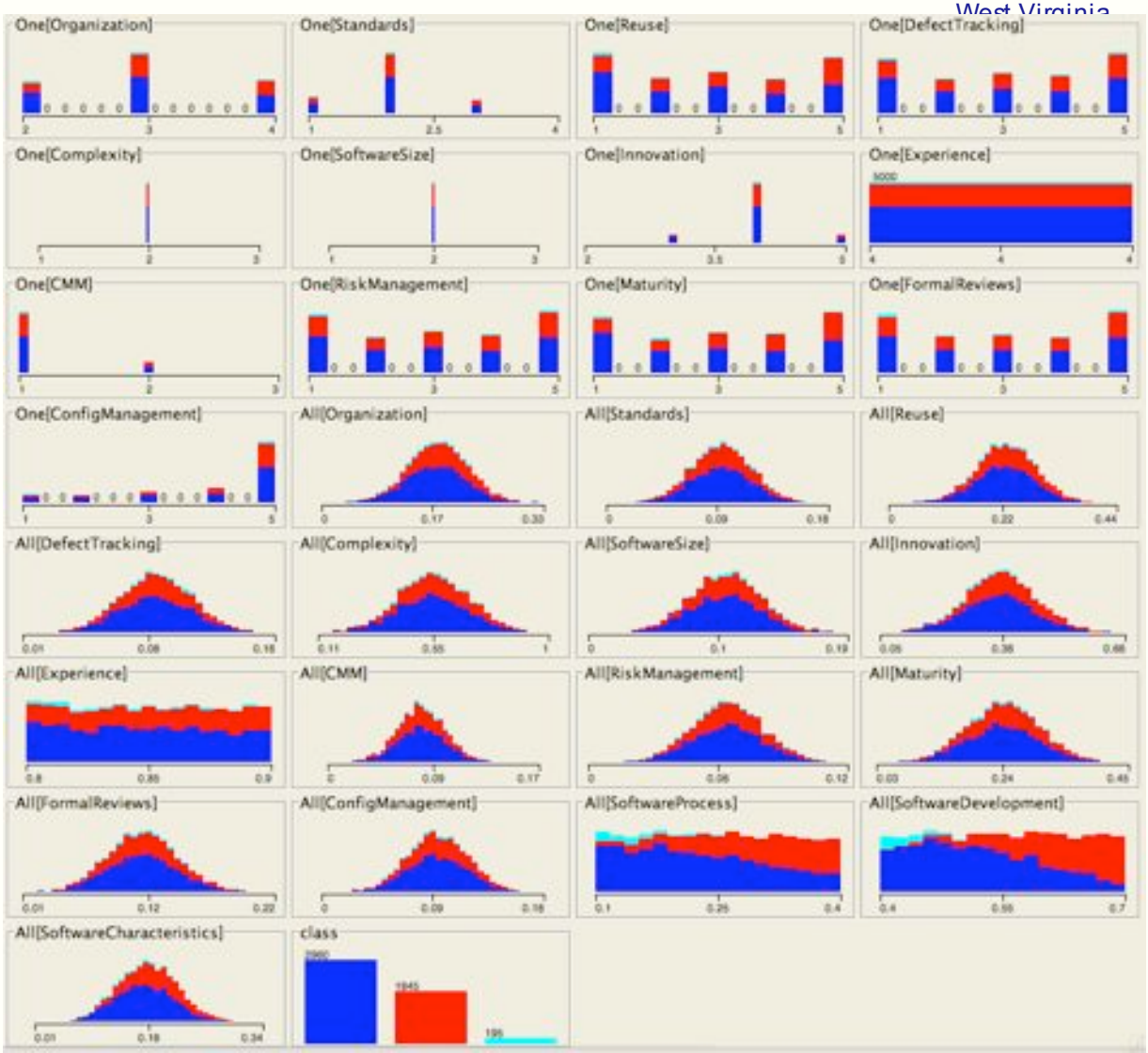
Warning: made up numbers!

```
BEGIN {
  to          = " to ";
  usually     = "0.1x";
  often       = "0.25x";
  sometimes   = "0.5x";
  rarely      = "0.75x";
}
```




```
function the(y) { return one(y) * all(y) }

function development() {
  return the("Experience") + the("Organization");}

function software()     {
  return the("Complexity")+the("Innovation")+ the("SoftwareSize")}

function process() {
  return the("Reuse") + the("Maturity") +
         the("CMM") + the("FormalReviews") +
         the("Standards") + the("ConfigManagement") +
         the("DefectTracking") + the("RiskManagement") +

function errorPotential() {
  return (all("SoftwareProcess") * process()) +
         (all("SoftwareDevelopment")* development()) +
         (all("SoftwareCharacteristics")* software())}
```

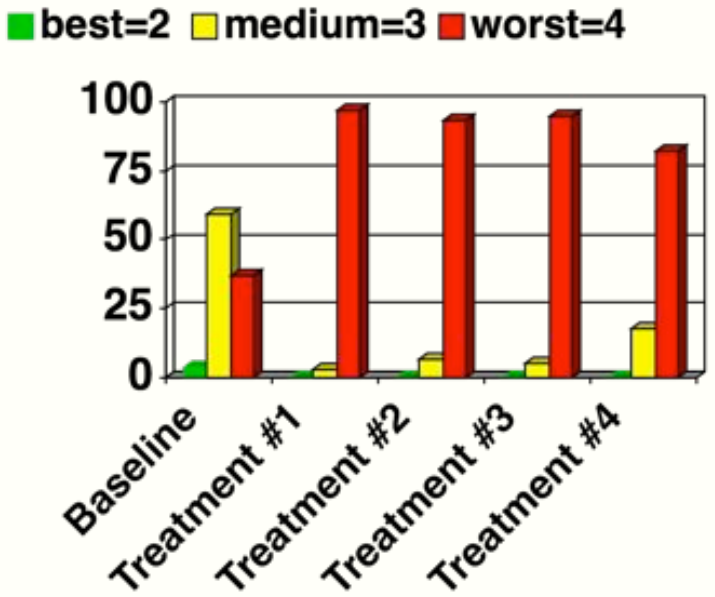
After 5000 runs

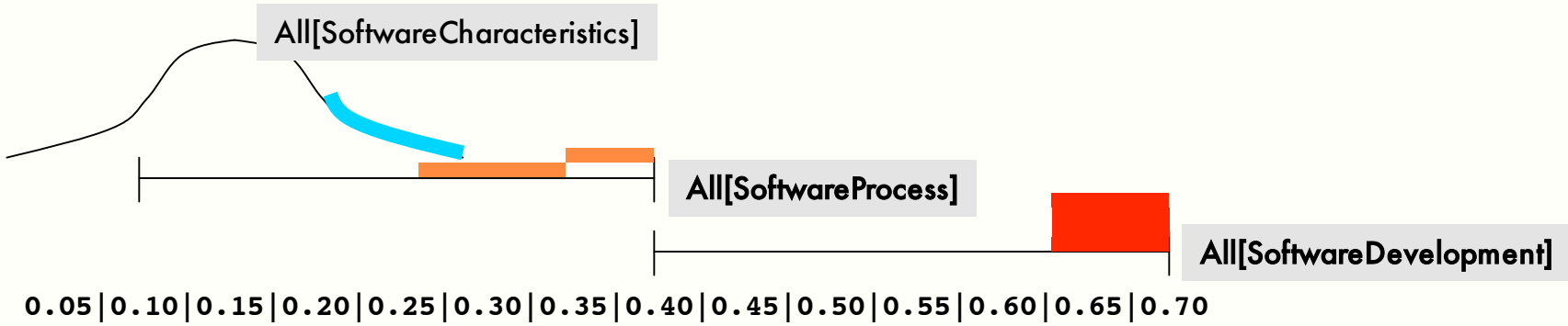


- **Lottsa data.**
- **What does it all mean?**
- **Q: What are the key patterns?**
- **A: Use the lift heuristic**

Treatment Learning results

- **Treatment #1 (top):**
  - All[SoftwareProcess] = 0.34 to 0.4
    and All[SoftwareDevelopment]= 0.64 to 0.7
- **Treatment #2:**
  - All[SoftwareCharacteristics]=0.21 to 0.34
    and All[SoftwareDevelopment]=0.64 to 0.7
- **Treatment #3:**
  - All[SoftwareDevelopment]= 0.64 to 0.7
    and All[SoftwareProcess]= 0.28 to ..0.34
- **Treatment #4:**
  - One[RiskManagement]=5
    and All[SoftwareDevelopment]=0.64 to 0.7





Research Heaven,
West Virginia

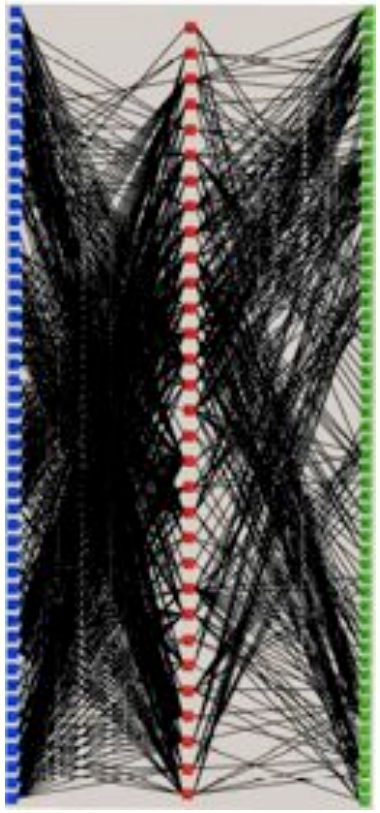
Case study 3

DDP: Cornford & Feather's requirements engineering language

(work with James Kiper, Jeremy Greenwald)

DDP

Research Heaven,
West Virginia



analyze this

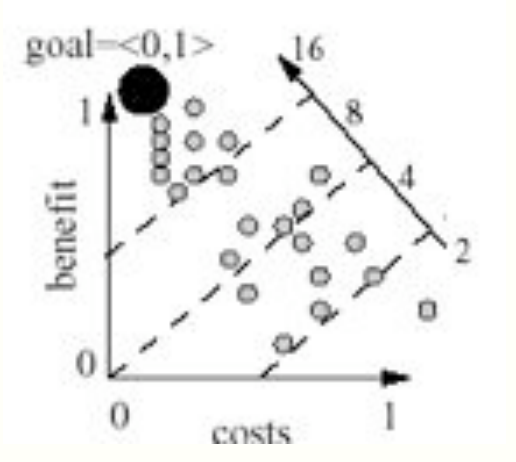
- **Cornford and Feather**
  - Visual tool for “group think”
  - RISKS hurt REQUIREMENTS
  - MITIGATIONS remove risks, cost money.
  - Seek cheap mitigations resolving risks that hurt the important requirements
- **Has been used for:**
  - Starlight, Deep Space 1&2, X2000 electronics packages; Interferometry design; Mars Globa Surveyor extended missions, Technology Infusion/Maturity assessments, ...
- **Being used for:**
  - SCrover: University of Southern California’s autonomous rover
  - Will be used for
  - Cost and risk models for autonomous systems

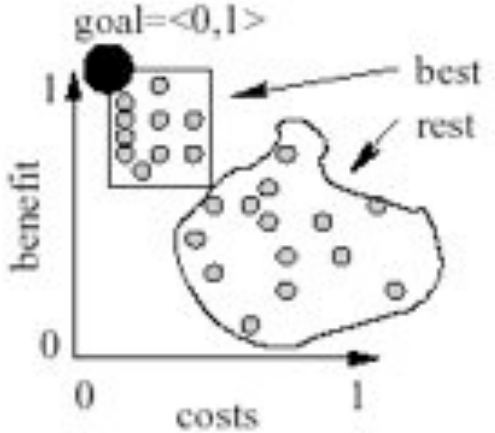
Optimizing DDP: learn better mitigations via via random mitigation selection

- **Use AI to find the cheapest mitigations that result in the greatest number of covered requirements**
  - Score1 = (benefit1 + cost1)/2
  - Benefit1 = benefit / maxBenefit
  - Cost1 = 1 - (cost /maxCost)
- **Methods:**
  - Simulated annealing (optimization method for non-linear systems)
    - Standard : use Score1
    - Regularized: Score1/(# actions +1)
  - Treatment learning with
    - D: diagonal classification
    - Bore: best or rest sampling
    - Surfer: a treatment learner decision support tool

- **Diagonal classification: N classes**

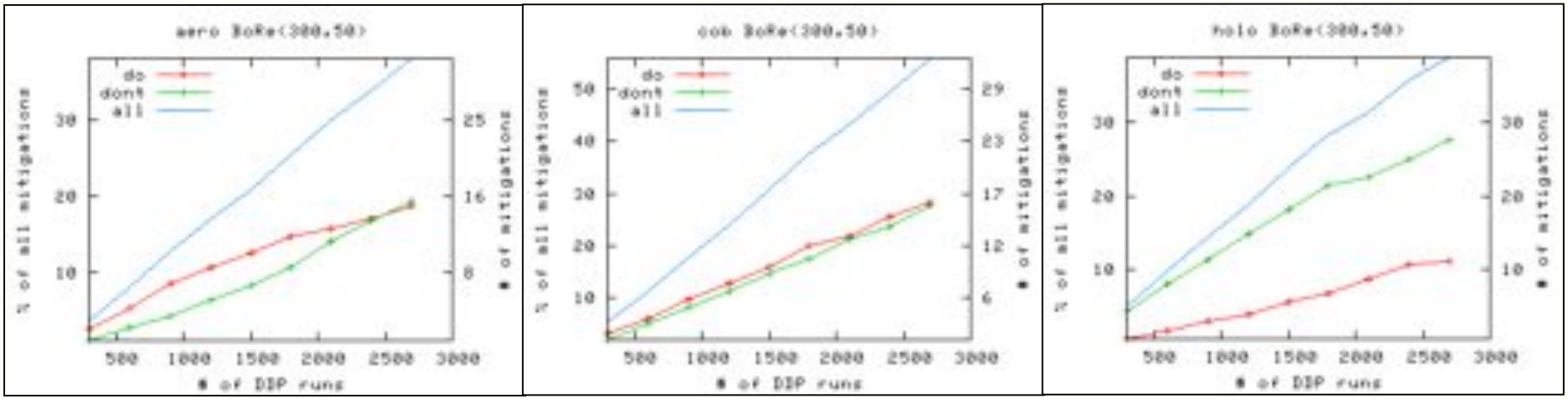
- **Bore: 2 classes**





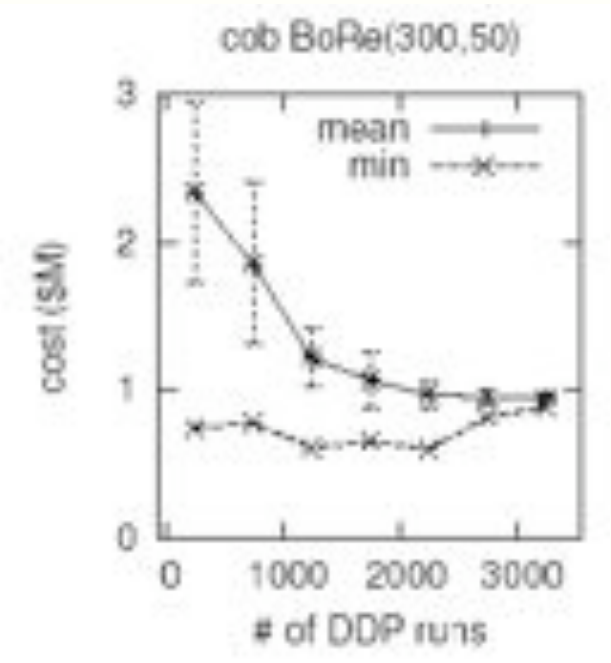
Surfer

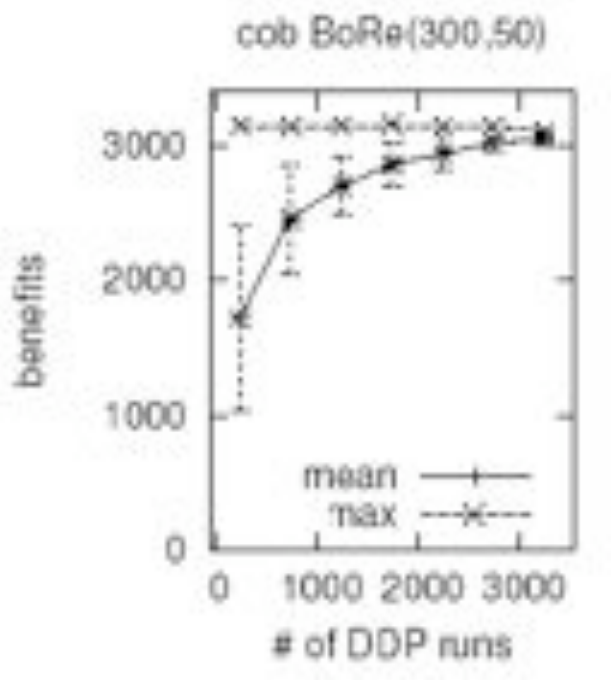
Currently, under the hood, SURFER calls treatment learning.
This may change…..



no more brittle point solutions

Research Heaven, West Virginia



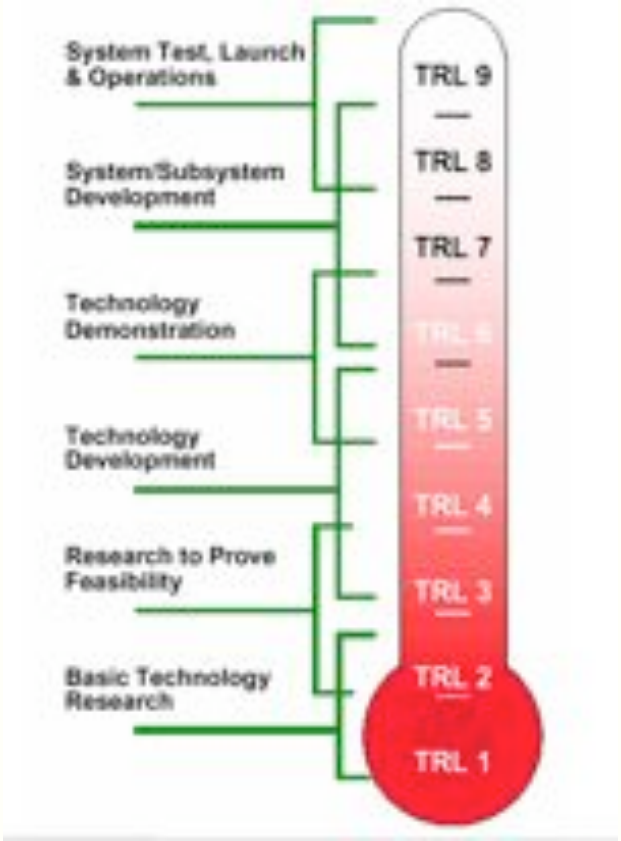


Research Heaven,
West Virginia

Enough with the case studies

Technology Readiness Level of the Work = 5

Research Heaven, West Virginia



- **5:**
  - Component and/or breadboard validation in a relevant environment

Potential Applications

Research Heaven,
West Virginia

- **Software process modeling when process details unclear**
- **Assemblies of systems from different developers**
- **Teams exploring options prior to large builds**
- **Decision making under uncertainty**
- **Simulation-based acquisition**
- **Solo developers**

Availability of data or case studies

Research Heaven,
West Virginia

DDP: see Martin Feather at JPL

SILAP: see Marcus Fischer at IVV

NEAR: http://menzies.us/pdf/05qrre.pdf

Barriers to research or applications

Research Heaven,
West Virginia

- Getting more examples
- Exploring option space is impossible without the options

Next Steps

Research Heaven,
West Virginia

- **More case studies**
  - SILAP: lots to do
  - Team X: excellent test bed
  - Synergy with HRT project on cost-benefits autonomous systems
- **Generalization**
  - N case studies
  - Reusable “marthas” extracted from the case studies
- **Better restraining policies**
  - Use internals of data miner to define what to try next
    - Bayesian analysis